B. Amendments to the Specification

[0050] A preferred, fixed scale, hardware platform 50 for the present invention is shown in Figure 2. The platform 50 is preferably implemented on a motherboard supporting the Intel® E7500 system control hub chipset 52, dual 2.2 GHz Intel® Xeon™ processors 54 (Intel Corporation, Santa Clara, California; www.intel.com), and a 1-Gbyte 200-MHz Double Data Rate (DDR) main memory array 56. The chipset 52 supports six PCI-X buses 58, individually capable of over 8-Gbps throughput and an aggregate throughput of at least 24-Gbps. A basic configuration of two 1-Gbps network interface controllers, supporting ingress and egress network connections, and one 10/100 Mbps network interface controller, supporting a management network connection, are connected to the PCI-X bus 58. A base configuration of three HiFn™ 7851 security processors 62 (Hifn, Inc., Los Gatos, California; www.hifn.com) provides hardware accelerated encryption and compression support for the generic data processing and control function of the processors 54. The security processors support symmetric programmable length block encryption algorithms, including 3-DES, at throughputs in excess of 400-Mbps per chip and programmable length block compression algorithms, including LZS, at throughputs in excess of 80MBps.

[0084] As represented in Figure 8A, the packet payload data of network file data packets corresponds to read and written portions of a file 220 recognized by a file system 34 ~~36~~. Individual packet payload data 222, generally as shown in Figure 8B, is preferably processed 194 into a sequence of logical access blocks $224_{1\text{-}N}$, as shown in Figure 8C with each logical access block containing a corresponding portion of the packet payload data 222. In an initial embodiment of the present invention, the file management header 226 is virtualized for all files associated with a real mount point and locally stored by the platform 50 effectively as part of the policy data held by the policy store 182. The applicable file management header is retrieved as part of the policy set applicable to the requested virtual mount point. The preferred embodiments of the present invention provide for the creation of a file management header 226 in connection with each Create file NFS/CIFS transaction. In one embodiment, the file management header 226 is created and written to the network storage resources 16 effectively as the first file data block as part of the creation of the file 220 on the network storage resources 16. One or more logical access blocks 224 can thereafter be appended to the file as created on the network storage

resources 16 and, subsequently, read and written in random order. Alternately, to optimize the storage and retrieval of data with respect to the network storage resources 16, individual or subsets of logical access blocks 224 and the file management header 226 can be written to separate I/O pages within the same or different file spaces and storage devices. In either case, in accordance with the present invention, qualified file data reads and writes directed to the network storage resources 16 are performed as discrete, logical access block-aligned transfers encompassing the offset and range of a client network file data request.

[0115] The preferred process 360 of performing an NFS/CIFS write request transaction is shown in Figure 12B. The received write file data request is received and processed 362 to expose the network control information 114. This information is then parsed 364 against the established policies 180, 182, with any compliance failures being reported 366 ~~386~~. The network control information 114 is then further processed 368 to identify the target file stored by the network storage resources 16, create and issue read requests to obtain the file meta-data, including the file management header 226. The logical access block offset and range are then determined 370, 372, adjusting as needed for the presence of the file management header 226 and compression of the logical access block 224 contained data. A file lock is asserted against the range logical access blocks $224_{A\text{-}X}$. The initial and terminal logical access blocks 224_A, 224_X are read 374 from the network storage resources 16, corrected 376 if the LAB ECC field 246 is present, decrypted 378, and decompressed 380, as needed. Integrity failure errors are reported 382. Data from the terminal logical access blocks 224_A, 224_X are merged 384 with the write data 342 and the combined data is resegmented 386, compressed 388 as appropriate, and encrypted 390. As applicable, LAB ECC values are computed and added 392 to the assembled 394 series of logical access blocks $224_{A\text{-}X}$. As the logical access blocks $224_{A\text{-}X}$ are assembled, one or more write network file data packets are constructed and sent to the network storage resources 16. Once the writing the logical access blocks $224_{A\text{-}X}$ has completed, the file lock is released.